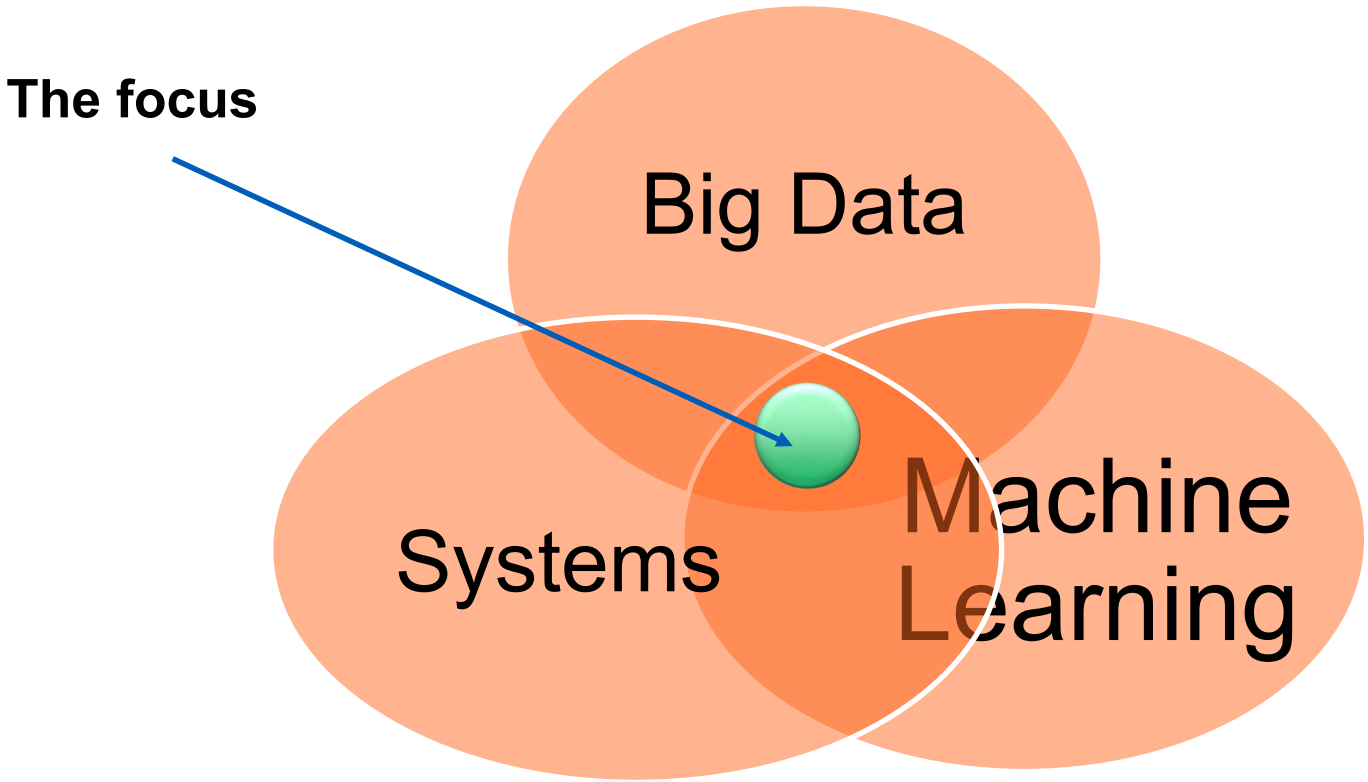
Aalto University
School of Science

Robustness, Reliability, Resilience, and Elasticity (R3E) for Big Data/Machine Learning Systems

Hong-Linh Truong
Department of Computer Science
linh.truong@aalto.fi, https://rdsea.github.io

Our focus in this course

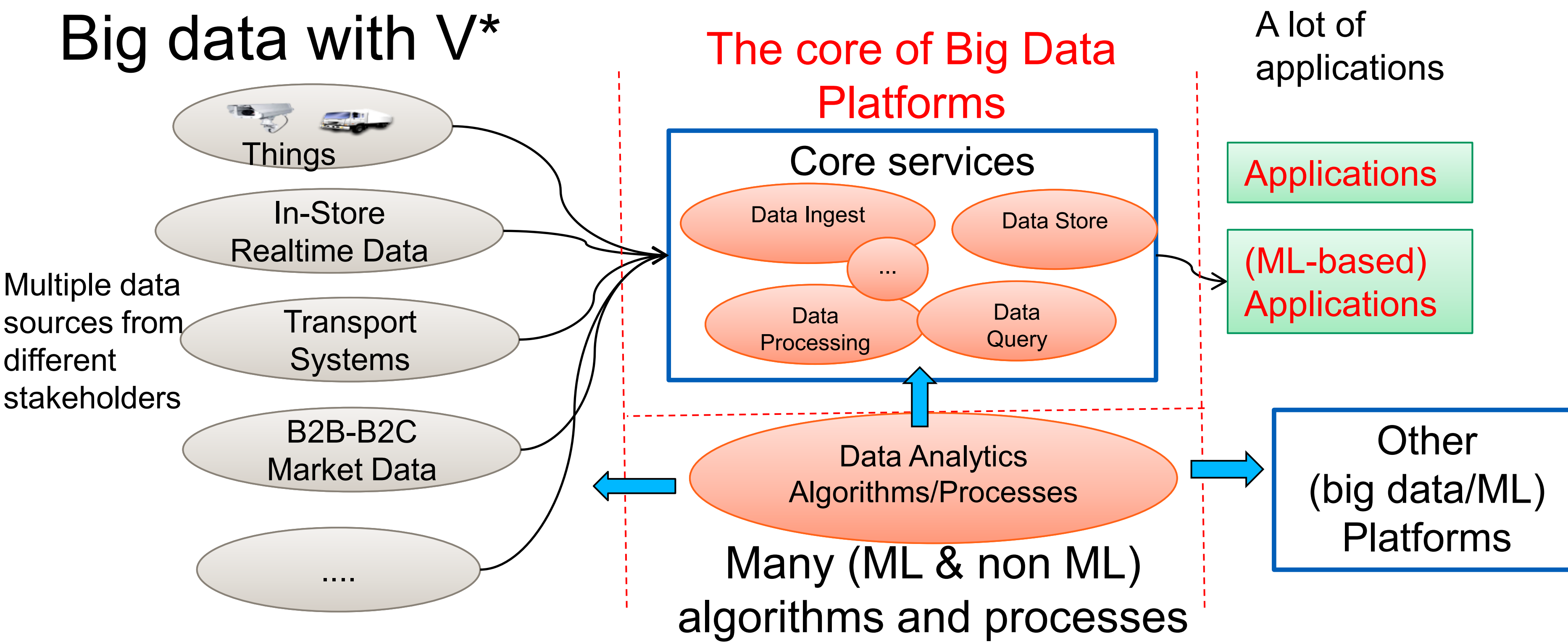


Learning objectives

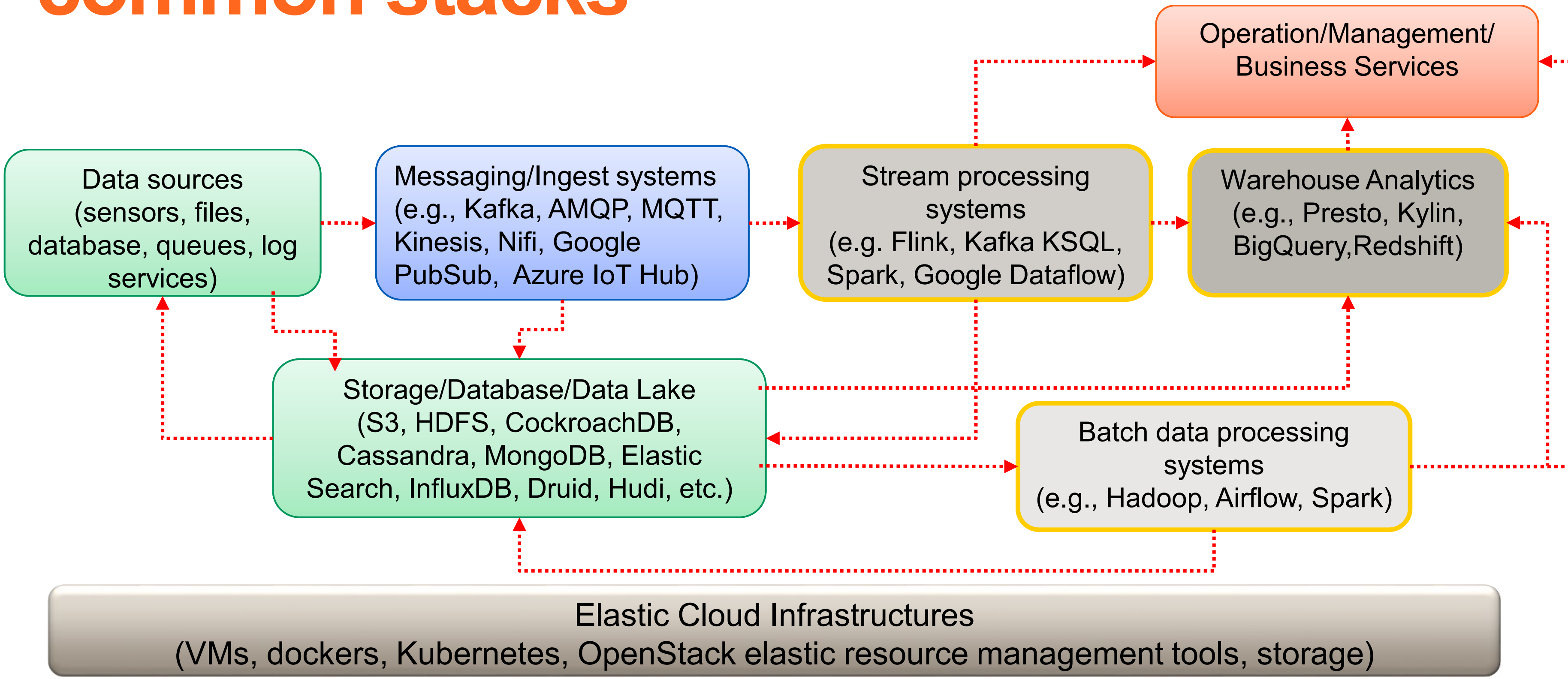
- **Identify and understand commonality and complexity in end-to-end Big Data/ML systems**
- **Understand design goals and concerns for robustness, reliability, resilience and elasticity of Big Data/ML systems**
- **Learn an elasticity-based approach for R3E**

Commonality and complexity in Big Data and Machine Learning systems

Big Data platforms: system of systems view



Big data at large-scale: example of common stacks



Examples from Big Data Platforms

Check our github:

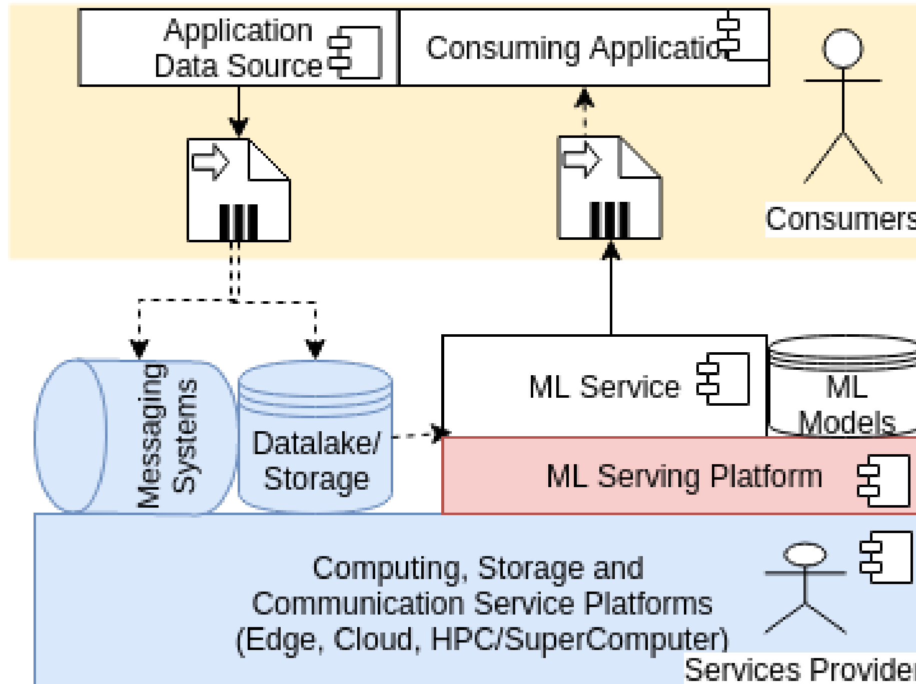
https://github.com/rdsea/bigdataplatforms

ML systems

- **Components in machine learning systems**
  - ML models are a kind of "data processing" programs
  - many other components for data preparation, data management, data movement, experiment management, and serving
- **Machine learning pipelines**
  - complex structured components, (meta)workflows
  - engineering tools: testing, monitoring, debugging, etc.
- **Data**
  - training/validation/test data, and data to be inferenced
  - models and parameters, experiment settings, and experiment data
  - from the big data platforms viewpoint: they are all data!

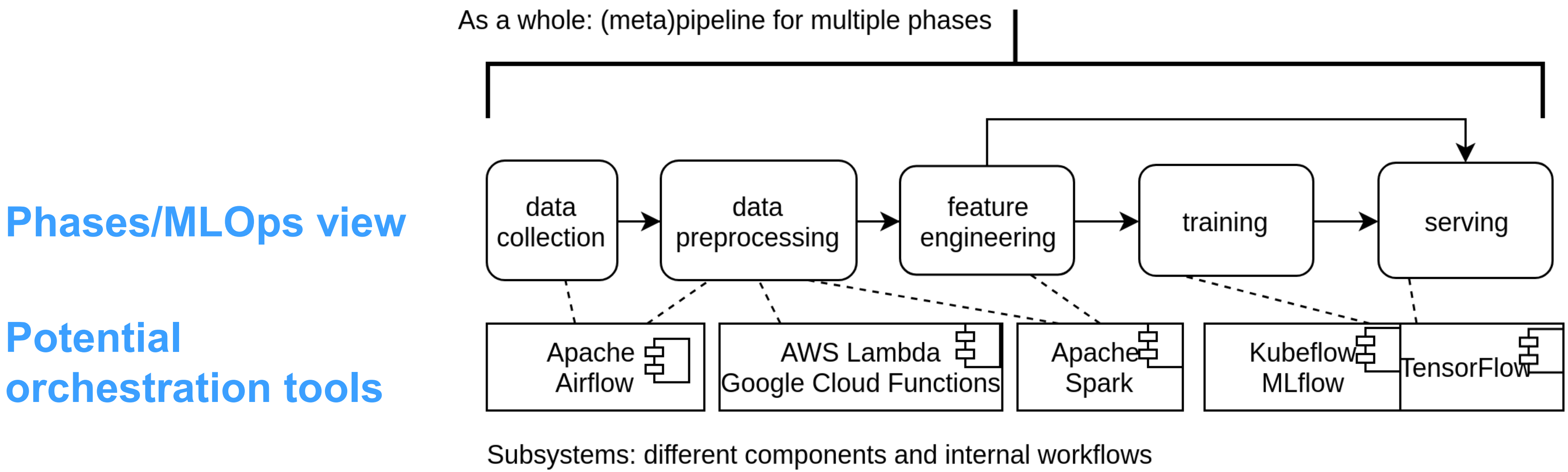
Consumers, model, services, platforms and infrastructures

- More than just ML models
- Complex design and components
- Mostly with distributed and high-performance computing systems



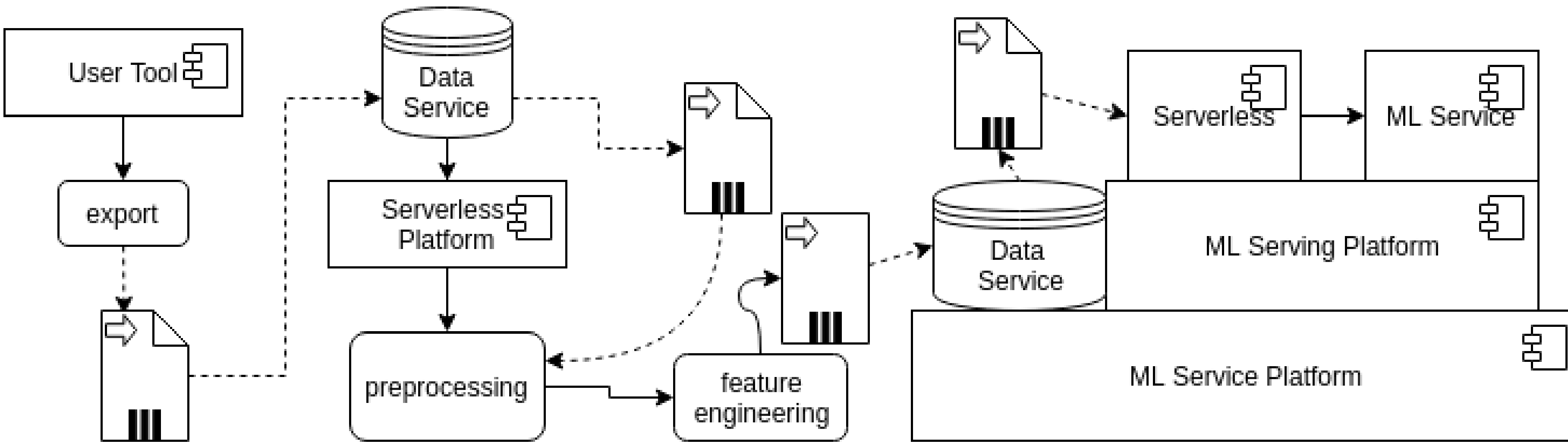
ML workflows

- **Two possible levels:**
  - meta-workflow or pipeline
  - inside each phase: pipeline/workflow or other types of programs
- **All are complex, executed in distributed systems**



An abstract example

Classifying objects in Building Information Model (BIM) in Architecture, Construction and Engineering



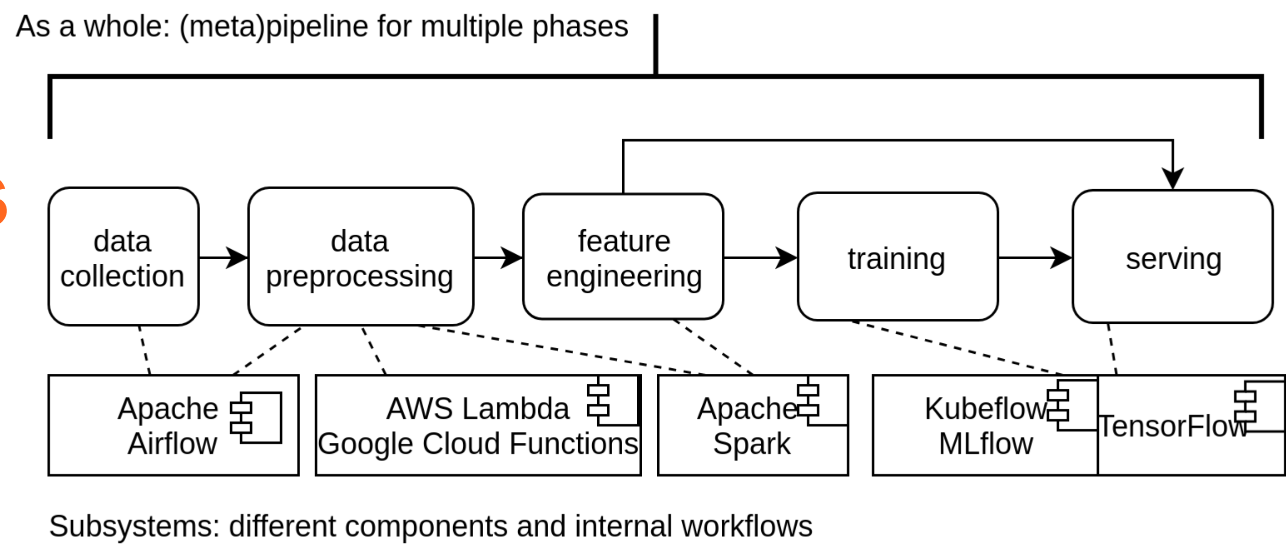
System view: common characteristics of big data and ML systems?

- **(Static) system structures and functions**
  - include software components, algorithms, input/output data
  - viewed as a whole, sub-systems, and individual parts

- **Computing and data infrastructures/platforms**
  - virtual machines/containers, brokers, storage, orchestration

- **Runtime quality/capability**
  - fault-tolerance, high-performance, high availability, secure, etc.

Examples of common components in big data and ML systems



Big data collection, ingestion, transformation

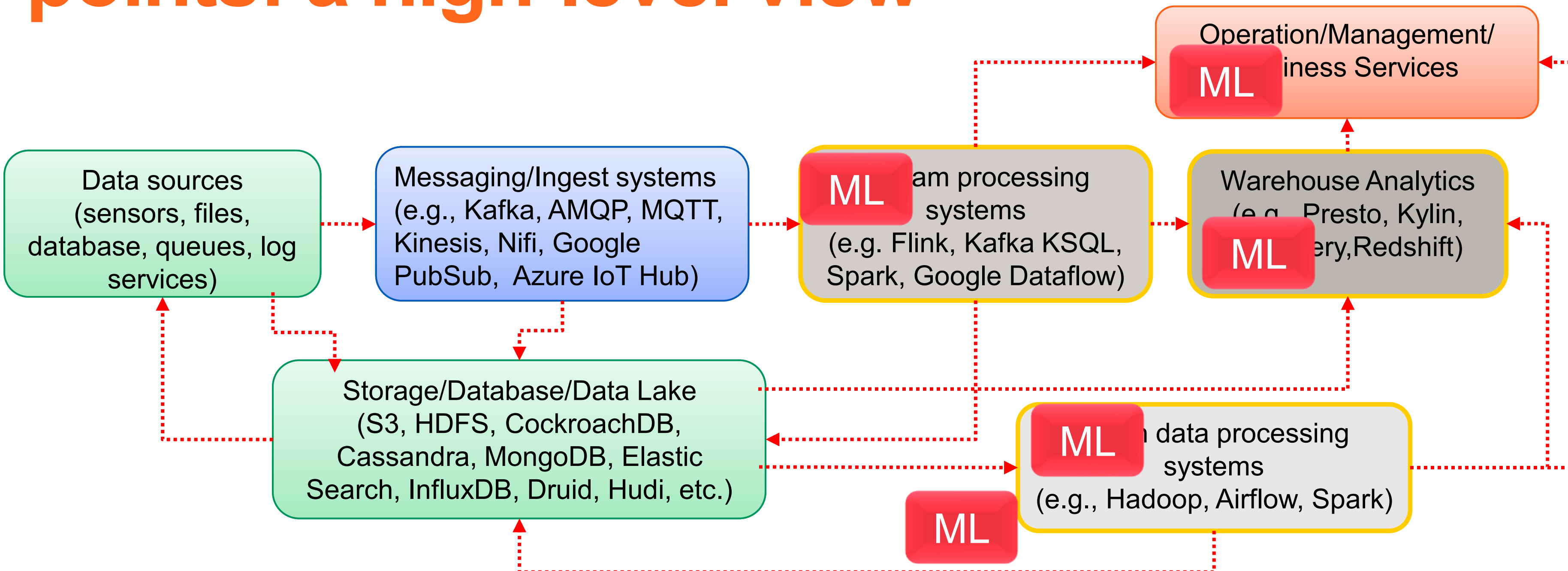
Big data processing

Resource management, workflow execution, data management tools, etc.

Integration between big data platforms and ML services

- **Data preparation**
  - For training purpose or for ML inference
  - Feature engineering, including feature store
  - Data quality/data concerns
- **Training**
  - Connecting big data sources to training processes
  - Distributed data movement and task execution
- **Serving**
  - Move data to serving processes

Data platforms and ML integration points: a high-level view



Real-world examples: Google Big Query ML, Tensorflow-IO, Azure Cosmos, mindsdb

Computing and data infrastructures

Cloud/HPC

- **Clusters of VMs/containers**
  - e.g., in Aalto we use CSC (https://www.csc.fi/)
- **High performance systems**
- **Known accelerators**
  - GPU and FPGA
- **New AI Accelerators/Processing Units**
  - TPU (Tensor Processing Unit)
  - Neutral Network Processor (NNP)
  - Vision Processor Unit (VPU)
  - IPU(Intelligent Processing Unit)

Google Cloud

amazon web services™

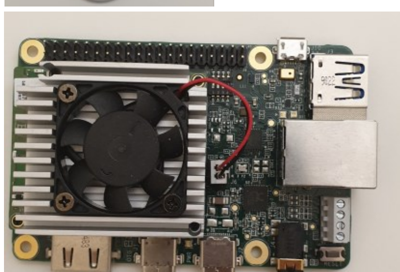
Illustration of the HPE Cray EX cabinets. Copyright: Hewlett Packard Enterprise

Figure source: https://www.lumi-supercomputer.eu/deep-dive-into-the-building-of-the-lumi-data-center/

Edge systems

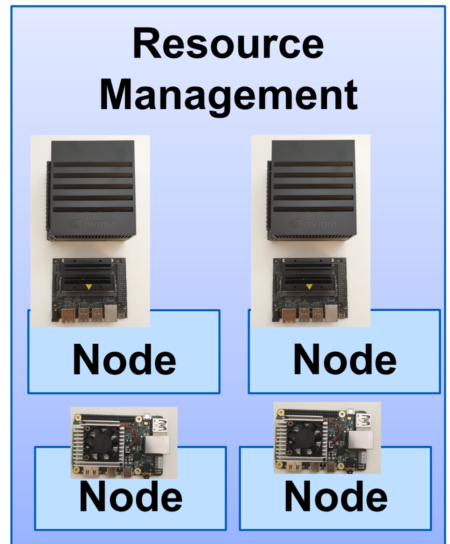
New types of edge and edge-cloud

Coral with Edge TPU System-on-Module, Google Edge TPU ML accelerator coprocessor



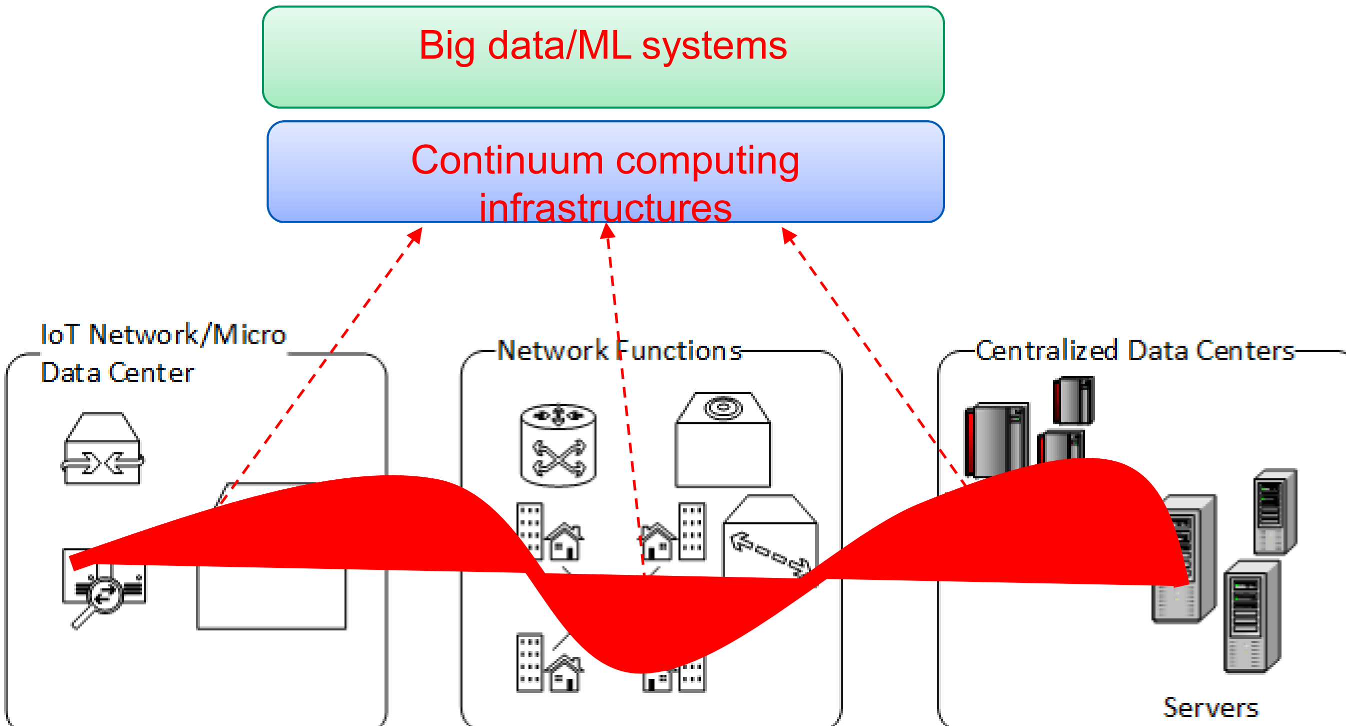
Jetson NVIDIA (GPU+CPU)

Distributed Edge systems



Our testbed for this course

Harnessing and orchestrating end-to-end resources across edge-cloud-HPC



Examples of common infrastructural/platform components

- **Data collection, ingestion, verification**
  - also data versioning management
- **Algorithms and serving components**
  - serving platforms and infrastructures
- **Configuration and workflow execution management**
- **Observability, monitoring and analysis**
- **Resource management and orchestration**

Runtime abilities/capabilities

Which runtime abilities/capabilities are important for your big data/ML systems?

Examples

- **Data/inference service performance**
- **ML model accuracy and data quality**
- **Cost**
- **Scalability**
- **Failure handle/incident management**
- **Site Reliability Engineering (SRE) concepts:**
  - Service level agreement (SLA), service level objective (SLO) and service level indicator (SLI)
    - *https://landing.google.com/sre/sre-book/toc/index.html*

Which runtime attributes are distinguishable in big data/ML systems, compared with that in common cloud services?

Robustness, Reliability, Resilience, and Elasticity (R3E)

Objectives for end-to-end Big Data/ML systems engineering

- **Deal with end-to-end aspects that the real world requires**
  - e.g., not just ML models and their optimization
- **Reduce software and data engineering effort**
- **Scale our systems**
  - big data, large-scale infrastructures and high number of customers/tenants
- **Optimize the system under various constraints**
- **Offer a production-level "reliable service" for customers/tenants**

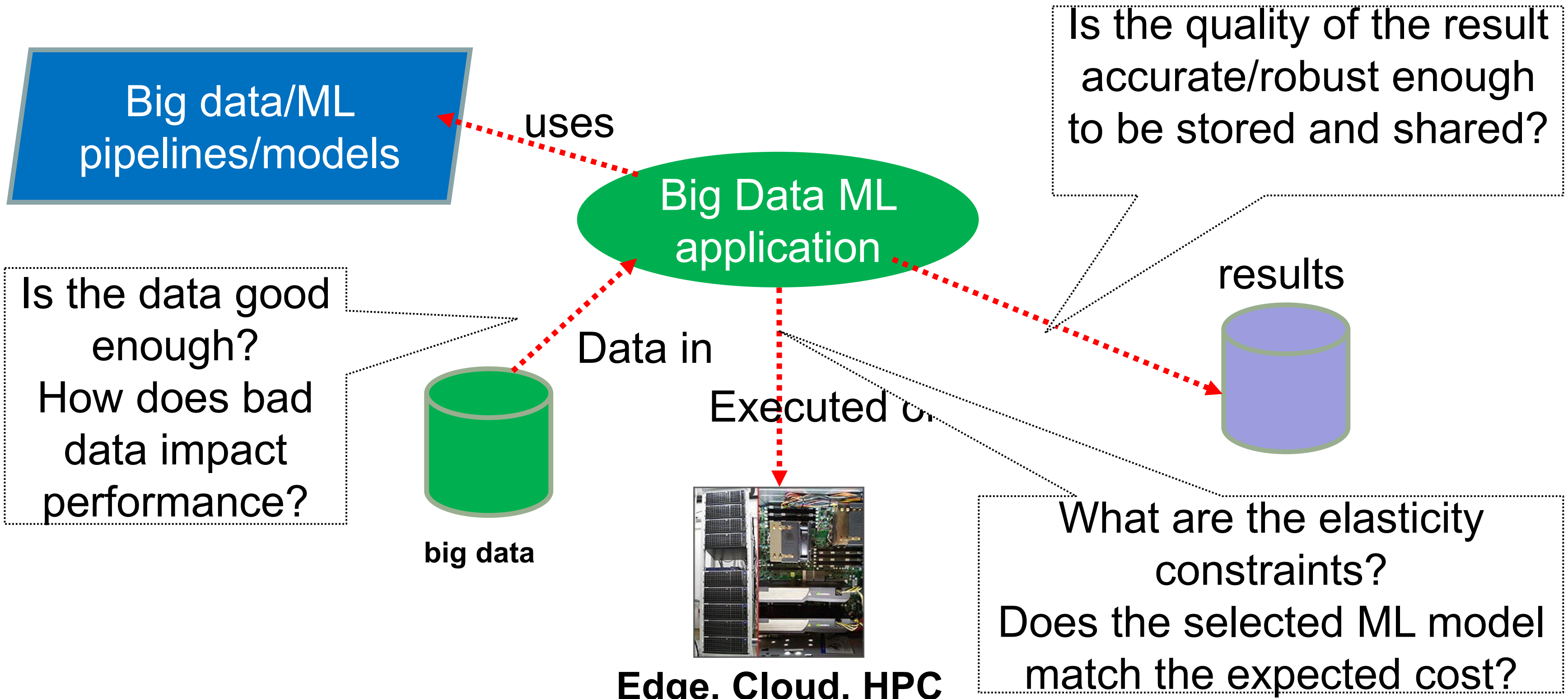
The complexity of end-to-end view

- **Engineering, optimizing and operating big data/ML systems**
  - which are key abilities that we should define, design, monitor, and measure?
  - how do we manage software artefacts, data, configuration, ...?
  - how to enable flexibility and execution management?
  - how to prepare for "future"/"emerging" infrastructures?
  - which are tools and frameworks that help reducing engineering complexity?

Key areas in our concerns

- **Software systems development**
  - testing, experimenting, benchmark, optimization, cost management
- **Resource management and adaptation**
  - execution atop multiple computing frameworks suitable for big data/ML, such as clouds, supercomputing, edges, …
- **(Runtime) ability/quality assurance**
  - specification, monitoring and assurance of performance, availability, costs, reliability, etc.

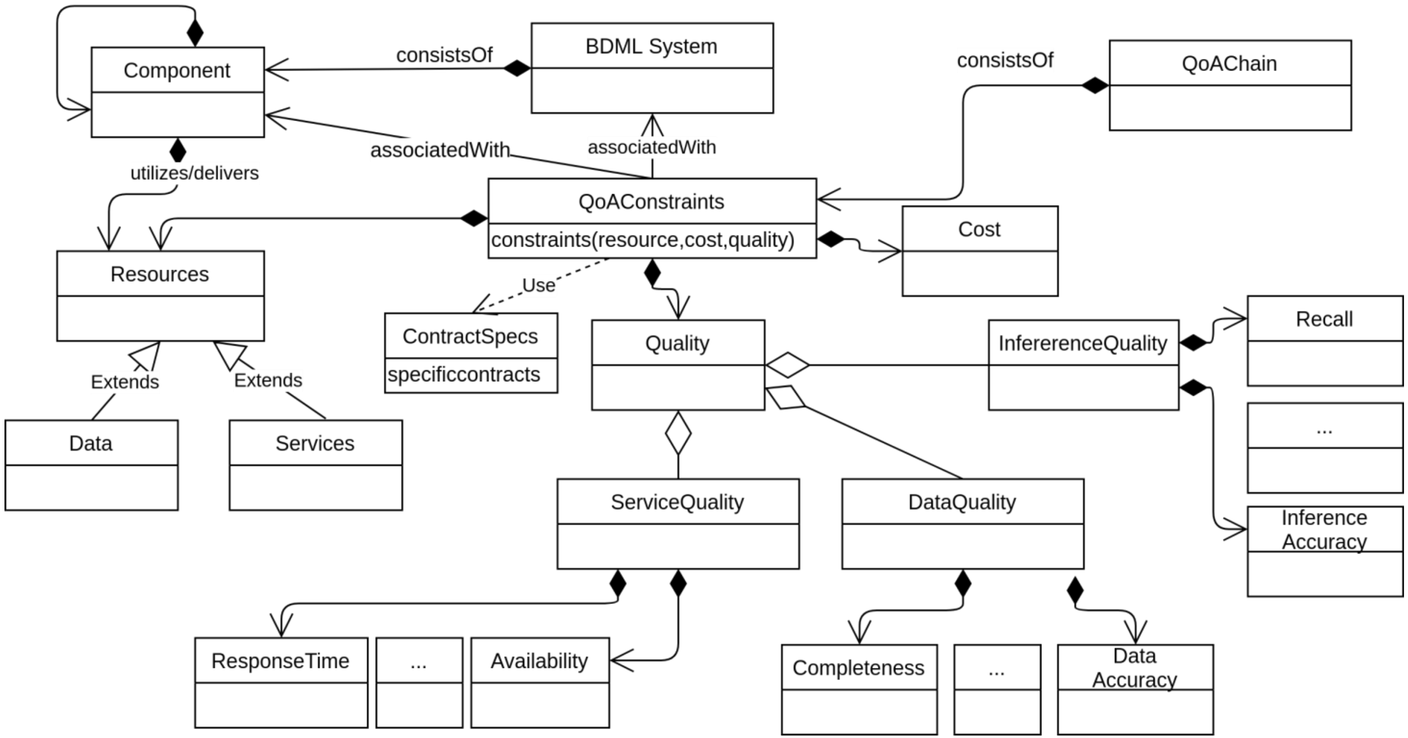
Quality of Analytics (QoA)



QoA = f (quality of result, performance, cost)

Key attributes/indicators

Just example, can be more!



Source: https://www.researchgate.net/publication/341762862_R3E_-An_Approach_to_Robustness_Reliability_Resilience_and_Elasticity_Engineering_for_End-to-End_Machine_Learning_Systems

Our focus – R3E

- **Robustness**
  - ability to cope with errors
- **Reliability**
  - ability to function according to the indented specification (in a proper way)
- **Resilience**
  - "ability to provide the required capability in the face of adversity"(https://www.sebokwiki.org/wiki/System_Resilience)
- **Elasticity**
  - ability to stretch and return to normal forms (under external forces)

Robustness

- **In ML**
  - overfitting
  - transfer learning
  - machine learning in an open-world
    - *how to deal with OOD (out-of-distribution) situations?*
  - when can we decide to stop training if performance/robustness does not improve? (or retrain when the performance/robustness degrades)
- **In Big Data**
  - how to deal with erroneous and bad data?

Reliability

- **System reliability versus "reliable service" (from customer/business/production view)**
- **System reliability**
  - reliable infrastructures, components, networks, …
- **"Reliable service" → reliable data analysis/inference**
  - without failure, with specified performance
- **Some hard problems**
  - have good and enough data, clean data
  - build pipelines without degraded performance and accuracy

Resilience

- **Common resilience issues in existing systems**
  - distributed software and systems bugs
  - system attacks
- **Some specific issues in big data/ML systems**
  - bias in data
  - well-known problems in adversary attacks in ML phases

Elasticity

- **Add and remove resources to tradeoff qualities**
  - CPUs, memory, data, networks, ...
- **Dynamic changes of algorithms**
- **Shift computation between edge and cloud infrastructures dynamically**
  - cloud data centers, edge systems and edge-cloud systems
- **Turn hyperparameter tradeoffs**

Short summary

Attributes	Cases from big data view	Cases from machine learning view
Robustness	deal with erroneous and bad data [48], data processing job robustness	dealing with imbalanced data, learning in an open-world (out of distribution) situations [36, 35, 23]
Reliability	reliable data sources, support of quality of data [49, 28], reliable data services [26], reliable data processing workflows/tasks [50]	reliable learning and reliable inference in terms of accuracy and reproducibility of ML models [35, 22]; uncertainties/confidence in inferences; reliable ML service serving
Resilience	software bugs, infrastructural resource failures, fault-tolerance and replication for data services and processing [47]	bias in data, adversary attacks in ML [25], resilience learning [14], computational Byzantine failures [8]
Elasticity	utilizing different data resources, increasing and decreasing data usage w.r.t. volume, velocity, quality; elasticity of underling resources for data processing [45]	elasticity of resources for computing [24, 21, 19], elasticity of model parameters; performance loss versus model accuracy; elastic model services for performance

Source: https://www.researchgate.net/publication/341762862_R3E_-An_Approach_to_Robustness_Reliability_Resilience_and_Elasticity_Engineering_for_End-to-End_Machine_Learning_Systems

Do we need to treat

Robustness, Reliability, Resilience, and Elasticity

equally in your design?

from which views?

An Approach with Elasticity Principles for R3E

Elasticity engineering

Designing and programming elastic components

Automatic deployment and configuration

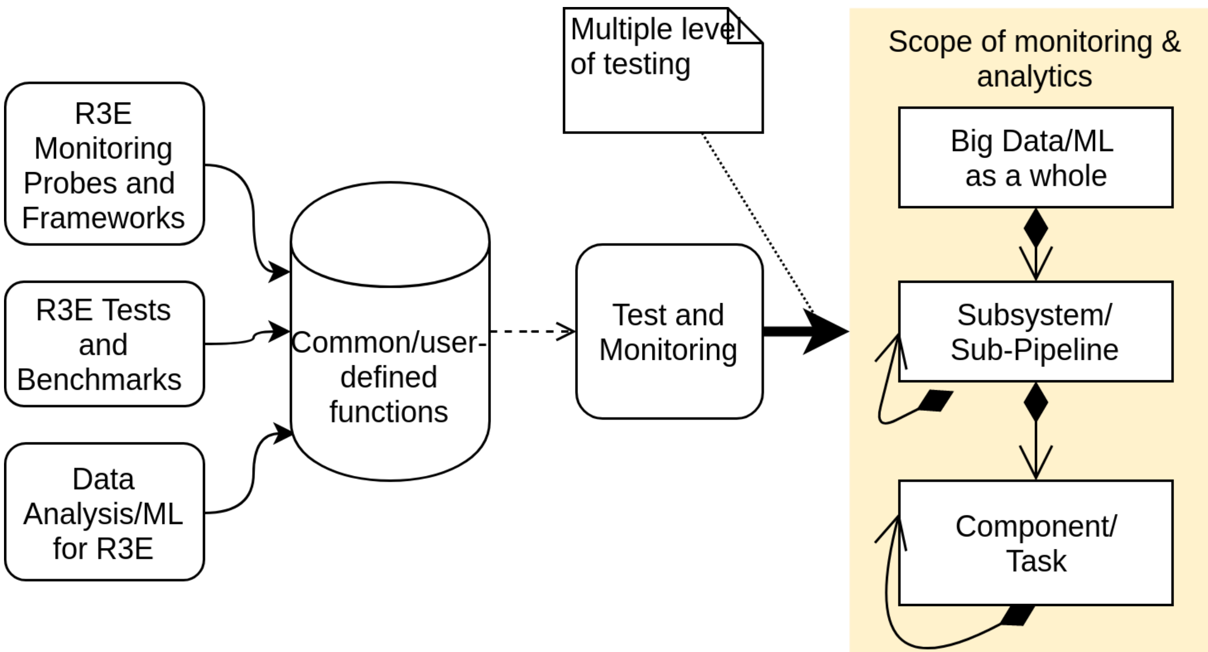
Elasticity monitoring and analysis

Coordinated elasticity control

Elasticity engineering for ML

- **Identifying, conceptualizing, and modeling elastic objects**
  - ML models, computing resources, data and QoA metrics
- **Defining and capturing elasticity primitive operations**
  - change resources, QoA metrics, model parameters, input data
- **Programming features for elastic objects**
  - with ML flows, coordinating QoA adjustment, dynamic serving models
- **Runtime deploying, control, and monitoring techniques for elastic objects**

Multi-level cross platforms monitoring and analysis



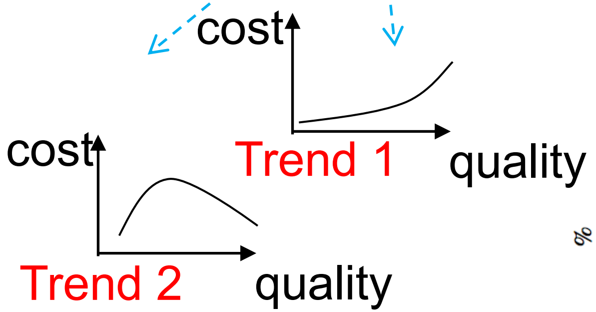
We will have a hands-on on observability and monitoring

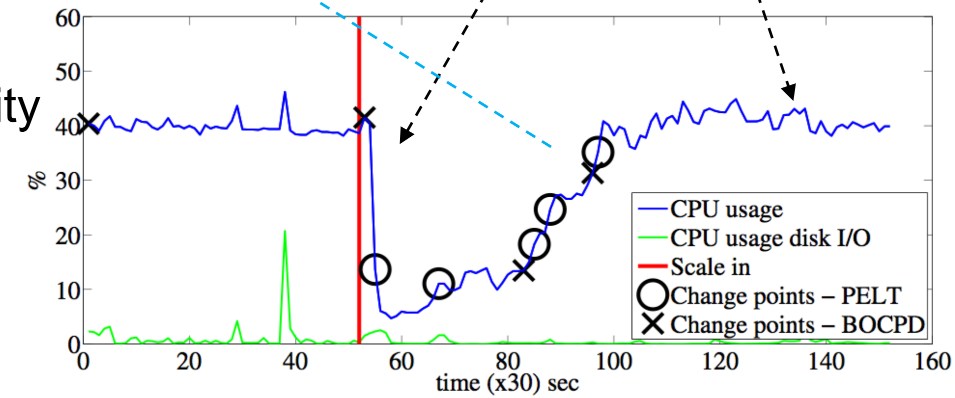
Detecting/learning elasticity

Elasticity Space Func (from when to when?)

Elasticity Trend Func

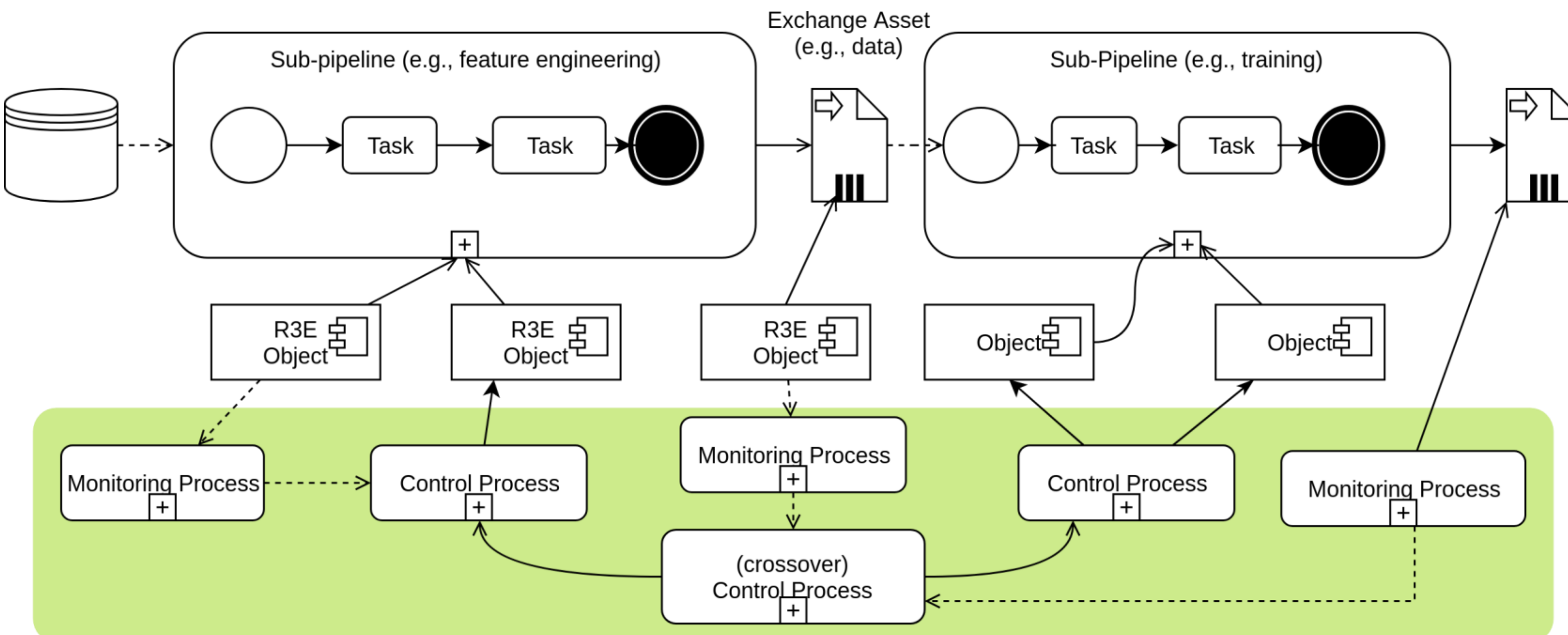
Change point detection algs (or ML-based)





Alessio Gambi, Daniel Moldovan, Georgiana Copil, Hong Linh Truong, Schahram Dustdar: On estimating actuation delays in elastic computing systems. SEAMS 2013: 33-42

Using control process to ensure QoA



Control process: can be AI/ML-based

Related hands-on: elastic ML serving

Beyond elasticity

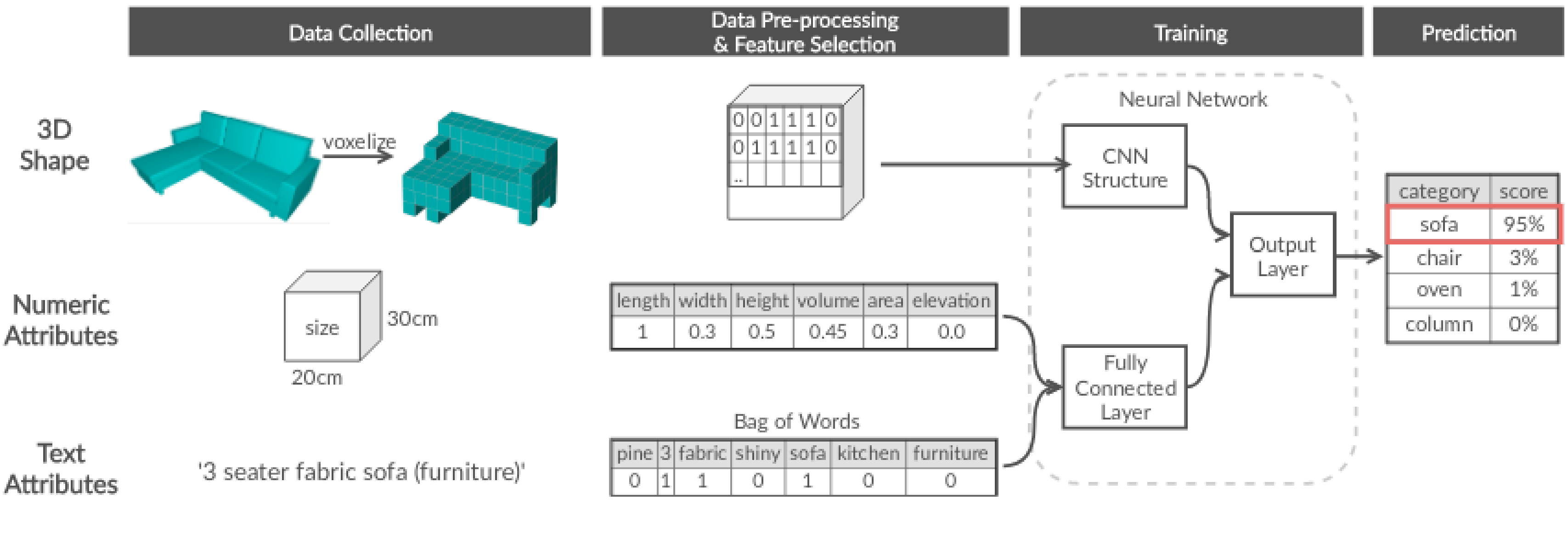
- **Can we apply similar techniques and methods for other attributes? E.g., resilience or reliability**

- **Specialization for some aspects**
  - Continuous ML model performance improvement (detect the accuracy degradation and retrain the model)
  - Performance of ML inferences and service with ensemble models elasticity and load balancing?

Aalto University
School of Science

Examples of ML systems built by our PhD/master students

ML classification for BIM (with Solibri data)

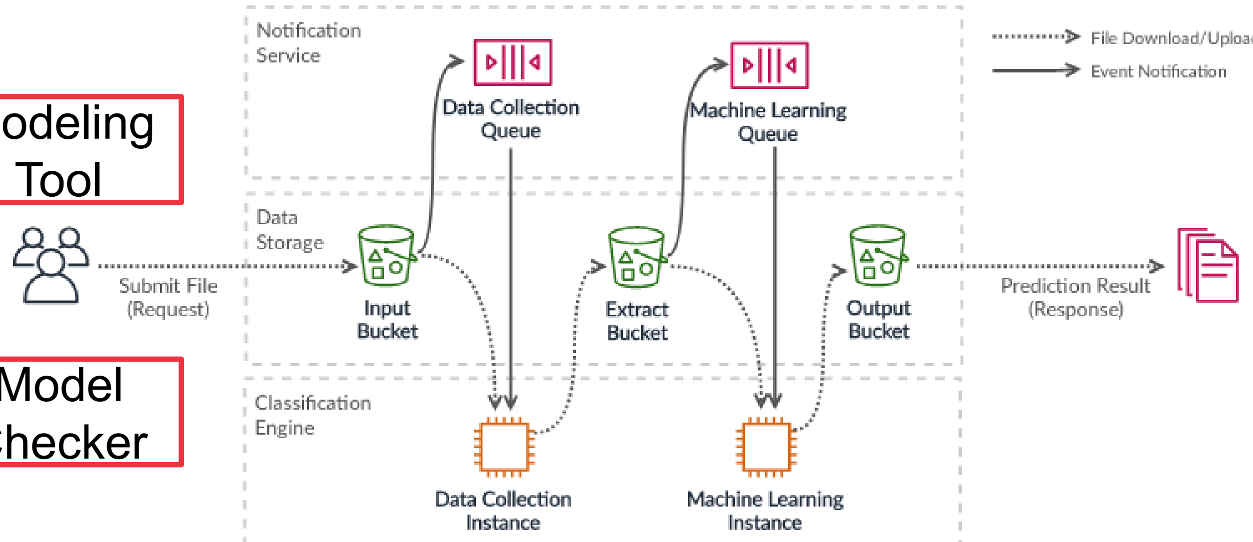


Source: Minjung Ryu, „Machine Learning-based Classification System for Building Information Models ", Aalto CS Master thesis, 2020
Ryu, M., Truong, HL. & Kannala, M. Understanding quality of analytics trade-offs in an end-to-end machine learning-based classification system for building information modeling. J Big Data 8, 31 (2021). https://doi.org/10.1186/s40537-021-00417-x

ML classification for BIM (with Solibri data)

Modeling Tool

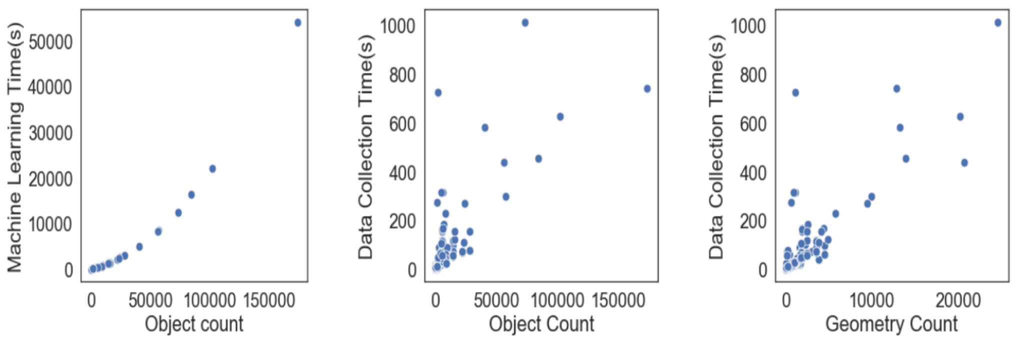
Model Checker



Source: Minjung Ryu, „Machine Learning-based Classification System for Building Information Models ", Aalto CS Master thesis, 2020
Ryu, M., Truong, HL. & Kannala, M. Understanding quality of analytics trade-offs in an end-to-end machine learning-based classification system for building information modeling. J Big Data 8, 31 (2021). https://doi.org/10.1186/s40537-021-00417-x

Results

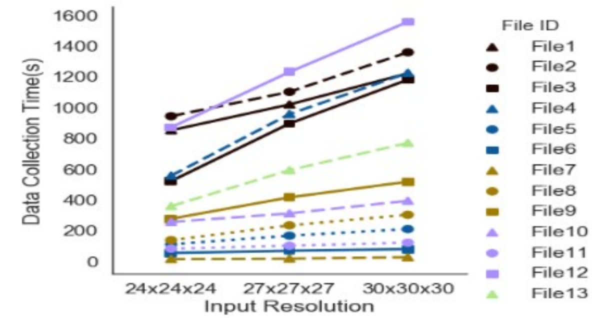
- **Data set: 591 classification cases from 146 models**
- **Machines: AWS/Local with/out GPUs**
- **Different cases and settings**



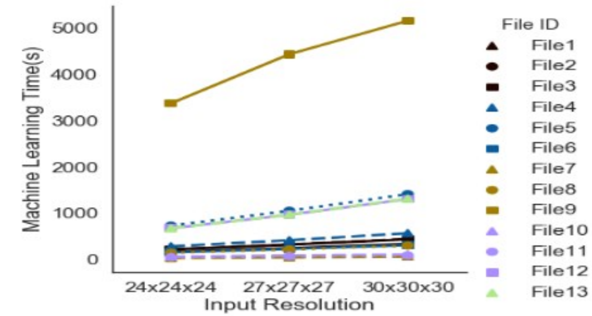
(a) Object count vs ML time (b) Object count vs DC time (c) Geometry count vs DC time

Figure 5: Impact of object counts on `DC time` and on `ML time`

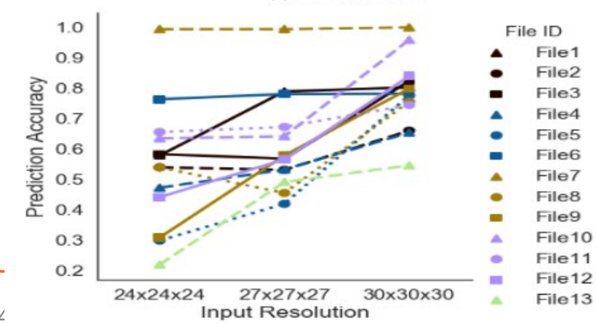
Reveal various relationships between types of data, extracting data resolution, machines and the accuracy of classifications



(a) Dim vs DC time



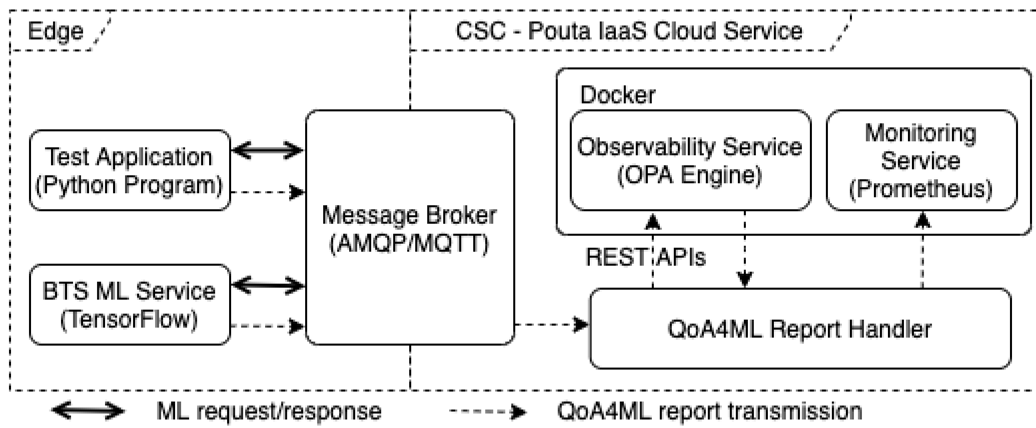
(b) Dim vs ML time



(c) Dim vs Accuracy

End-to-end edge-cloud ML serving

- **Dynamic inferences of load of power grid using LSTM, TensorFlow**
  - IoT data from Base Transceiver Station (BTS)
- **Training in cloud and export to the edge (BTS-model-edge) and retraining several times in the cloud (BTS-model-cloud)**
- **Deployment**
- **Contracts:**
  - ResponseTime
  - Inference Accuracy
  - Data Quality

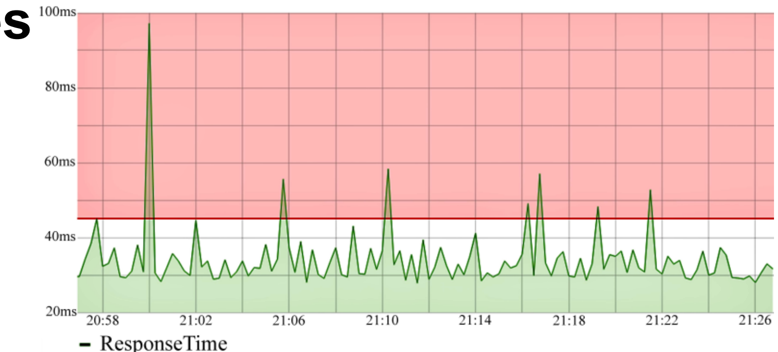


Effect of edge and cloud serving platform deployment

Both consumer and service are in the same edge; 3000 records per 15 minutes

Broker Deployment	Broker Type	Violation Rate
Edge (Raspberry PI, local network)	MQTT	12%
	AMQP	8%
CSC - Cloud	MQTT	41%
	AMQP	16%

Broker is in the cloud

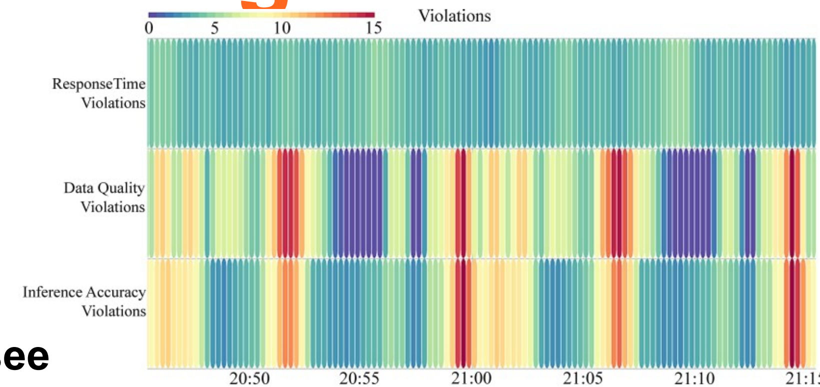


Both consumer and broker are in the same edge

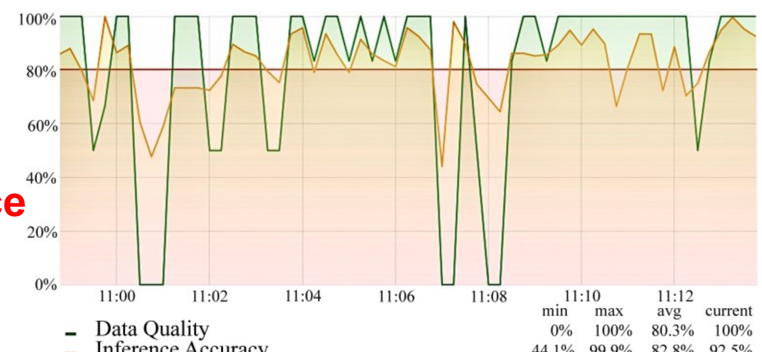
ML Service Deployment	Broker Type	Violation Rate
Edge (container, Google Cloud)	MQTT	20%
	AMQP	18%
CSC - Cloud	MQTT	38%
	AMQP	21%

Source: Hong-Linh Truong, Minh-Tri Nguyen, QoA4ML – A Framework for Supporting Contracts in Machine Learning Services, 2021 IEEE International Conference on Web Services (ICWS), September 5-10, 2021.

Data and inference accuracies monitoring



Help to see correlations among attributes: data quality and inference accuracy



Help to detect outdated models in ML services: violation changes when retraining models

Time Period	Violation Rate (training once)	Violation Rate (training every 2-hour)
0am-2am	2%	2%
2am-4am	4%	3%
4am-6am	18%	4%
6am-8am	15%	6%
8am-10am	10%	4%
10am-12pm	7%	4%
12pm-2pm	3%	3%
2pm-4pm	5%	5%
4pm-6pm	10%	7%
6pm-8pm	15%	5%
8pm-10pm	4%	2%
10pm-0am	1%	3%

Study log for this week

Think about

- **What does it mean R3E for** *YOUR big data and machine learning systems*?
- **Read one of the papers in the reading list for today lecture**

Then

- **in your experience/work, which ones of R3E concern you most? Why? What would you do? What do you look for?**
- **~1-2 page – submit it to the MyCourses for comments/feedback (keep it in your git)**

Thanks!

Hong-Linh Truong
Department of Computer Science

rdsea.github.io